



\*\*FILE\*\*ID\*\*DBGSTO

G 1

DDDDDDDDDD	BBBBBBBBBB	GGGGGGGGGG	SSSSSSSS	TTTTTTTTTT	000000	
DDDDDDDDDD	BBBBBBBBBB	GGGGGGGGGG	SSSSSSSS	TTTTTTTTTT	000000	
DD DD	BB BB	GG	SS	TT	00	00
DD DD	BB BB	GG	SS	TT	00	00
DD DD	BB BB	GG	SS	TT	00	00
DD DD	BB BB	GG	SS	TT	00	00
DD DD	BBBBBBBBBB	GG	SSSSSS	TT	00	00
DD DD	BBBBBBBBBB	GG	SSSSSS	TT	00	00
DD DD	BB BB	GG GGGGGG	SS	TT	00	00
DD DD	BB BB	GG GGGGGG	SS	TT	00	00
DD DD	BB BB	GG GG	SS	TT	00	00
DD DD	BB BB	GG GG	SS	TT	00	00
DDDDDDDDDD	BBBBBBBBBB	GGGGGGGG	SSSSSSSS	TT	000000	....
DDDDDDDDDD	BBBBBBBBBB	GGGGGGGG	SSSSSSSS	TT	000000	....

LL		SSSSSSSS
LL		SSSSSSSS
LL		SS
LLLLLLLLLL		SSSSSSSS
LLLLLLLLLL		SSSSSSSS

```
1 0001 0 MODULE DBGSTO ( IDENT = 'V04-000' ) =
2 0002 1 BEGIN
3
4 0004 1 ****
5 0005 1 *
6 0006 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
7 0007 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
8 0008 1 * ALL RIGHTS RESERVED.
9
10 0010 1 *
11 0011 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
12 0012 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
13 0013 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
14 0014 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
15 0015 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
16 0016 1 * TRANSFERRED.
17 0017 1 *
18 0018 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
19 0019 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
20 0020 1 *
21 0021 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
22 0022 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
23 0023 1 *
24 0024 1 *
25 0025 1 ****
26 0026 1 *
27 0027 1 FACILITY: DEBUG
28 0028 1 ++
29 0030 1 FUNCTIONAL DESCRIPTION:
30 0031 1 DECLARES GLOBAL VARIABLES FOR DEBUG FACILITY
31
32 0032 1 *
33 0033 1 Version: 1.12
34 0034 1 *
35 0035 1 History:
36 0036 1 Author: Carol Peters, 03 Jul 1976: Version 01
37 0037 1 *
38 0038 1 Modified by:
39 0039 1 Bruce Olsen, 11 SEP 1979
40 0040 1 Ken Nappa, 28 APR 1980
41 0041 1 Richard Title, 21 AUG 1981
42 0042 1
43 0043 1
44 0044 1 Revision history:
45 0045 1 3.00 21-AUG-81 RT Added some globals that are used during the
46 0046 1 source line display.
47 0047 1 3.01 25-Sep-81 RT Added & to dbg$token_table
48 0048 1 3.02 20-Oct-81 RT Added dbg$gb_search_ptr and dbg$gb_def_search
49 0049 1 to implement the SEARCH command.
50 0050 1 3.03 12-Nov-81 RT Added dbg$gl_nest_stack and dbg$gl_nest_level
51 0051 1 to fix a bug in handling nested subscript expressions
52 0052 1 in FORTRAN and BASIC.
53 0053 1 3.04 20-Nov-81 RT Added dbg$gb_set_module_flag. This is used to
54 0054 1 allow for module names that begin with a number.
55 0055 1 3.05 21-Dec-81 RT Added DBGSGB_EXC_BRE_FLAG and DBGSGB_GO_ARG_FLAG
56 0056 1 to handle continuing from an exception break.
57 0057 1 3.06 21-Dec-81 RT Deleted changes 1.01 through 1.10 from this list
```

```
58 0058 1
59 0059 1 | 3.07 3-May-82 JF    Added DBG$GB IMPLEMENTATION
60 0060 1 | 3.08 06-May-82 RT    Added data structure for SET DEFINE
61 0061 1 | 4.0 31-Aug-83 PS    Added read error count global variable
62 0062 1 | 4.01 02-Sep-83 WC3    Added DBG$GL_CURRENT_PRIMARY
63 0063 1 | 4.02 13-Oct-83 WC3    Added DBG$GL_MOVED_DST_LIST_HEAD
64 0064 1 |
65 0065 1 |
66 0066 1 ! INCLUDE FILES
67 0067 1 |
68 0068 1 |
69 0069 1 REQUIRE 'src$:DBGPROLOG.REQ';
70 0203 1 LIBRARY 'LIBS:DBGGEN.L32';
71 0204 1
72 0205 1 ++
73 0206 1 ****
74 0207 1 NOTE:
75 0208 1
76 0209 1 ALL initialization of addresses and pointers in the debugger
77 0210 1 MUST be done dynamically
78 0211 1 to maintain position independence. At compile time these addresses
79 0212 1 are relative to 0, but are relocated at run time since the debugger
80 0213 1 is brought in "behind" the user program.
81 0214 1 ****
82 0215 1 --
83 0216 1
84 0217 1 !GLOBAL LITERAL
85 0218 1 The base of RST storage. Some of the RST data structures base
86 0219 1 'pointers' off this. This is now a link time input
87 0220 1
88 0221 1 DBGS_RST_BEGIN = %X'7FFF0000'; for standard and test debugger
89 0222 1 DBGS_RST_BEGIN = "somewhere in P0 for SUPERDEBUG
90 0223 1
91 0224 1
92 0225 1 EXTERNAL LITERAL DBG$GL_SUP_OR_TEST : WEAK;
93 0226 1
94 0227 1 GLOBAL DBG$GV_CONTROL : VECTOR[2,BYTE] INITIAL(
95 0228 1   WORD(
96 0229 1     (DBG$GL_SUP_OR_TEST) OR
97 0230 1     (1^B)));
98 0231 1
99 0232 1 ++
100 0233 1 THE dbg$char_table associates each ASCII character with a value
101 0234 1 from 0 to n. The meaning of the numeric value can be seen in
102 0235 1 literal definitions declared in SCALIT.BEG (for example, 1 is bound
103 0236 1 to alpha).
104 0237 1 --
105 0238 1 GLOBAL BIND
106 0239 1    dbg$char_table = UPLIT BYTE (
107 0240 1
108 0241 1    6. 0. 0. 0. 0. 0. 0. 0. !000-007    treat null char as lf
109 0242 1    0. 4. 6. 6. 6. 6. 0. 0. !010-017    tab, lf, vtab, ff, cr
110 0243 1    0. 0. 0. 0. 0. 0. 0. 0. !020-027
111 0244 1    0. 0. 0. 0. 0. 0. 0. 0. !030-037
112 0245 1    4. 5. 16. 28. 1. 29. 0. 16. !040-047    space ! " # $ % & '
113 0246 1    9. 10. 21. 11. 24. 12. 20. 13. !050-057    ( ) * + - /
114 0247 1    2. 2. 2. 2. 2. 2. 2. 2. !060-067    0 1 2 3 4 5 6 7
```

```
115      0248 1    2, 2, 14, 15, 22, 25, 23, 0, 070-077     8 9 : < = > ?
116      0249 1    16, 3, 3, 3, 3, 3, 3, 1, 100-107     A B C D E F G
117      0250 1    1, 1, 1, 1, 1, 1, 1, 1, 110-117     H I J K L M N O
118      0251 1    1, 1, 1, 1, 1, 1, 1, 1, 120-127     P Q R S T U V W
119      0252 1    1, 1, 1, 1, 26, 18, 27, 17, 1, 130-137     X Y Z \ J ^ g
120      0253 1    0, 8, 8, 8, 8, 8, 8, 7, 140-147     a b c d e f g
121      0254 1    7, 7, 7, 7, 7, 7, 7, 7, 150-157     h i j k l m n o
122      0255 1    7, 7, 7, 7, 7, 7, 7, 7, 160-167     p q r s t u v w
123      0256 1    7, 7, 7, 0, 0, 0, 0, 0, 170-177     x y z { : ) ^ delete
124      0257 1
125      0258 1    ) : VECTOR [,BYTE];
126      0259 1
127      0260 1    ! These two globals were copied over from DBGBLI.B32 after that module
128      0261 1    was eliminated.
129      0262 1
130      0263 1    GLOBAL
131      0264 1    dbg$access_list : VECTOR [6] INITIAL (REP 6 OF (0)); ! access actuals needed for structure
132      0265 1    ! references. The first element will
133      0266 1    ! contain the no. of actuals
134      0267 1
135      0268 1    GLOBAL
136      0269 1    dbg$gl_modrstptr2; ! Holds module rst pointer during TYPE command.
137      0270 1    ! e.g., TYPE MOD1\10,20,30
138      0271 1
139      0272 1    GLOBAL
140      0273 1    ++
141      0274 1    ! Byte vectors are used to contain the
142      0275 1    ! 'mode', 'step type', 'search type',
143      0276 1    ! and output configuration data structures and
144      0277 1    ! also, the buffers used by RMS to return fully qualified filespecs
145      0278 1    !--
146      0279 1    dbg$gb_def_mod: VECTOR [mode_levels * mode_lvl_size, BYTE], ! DEFAULT MODE BLOCK
147      0280 1    DBG$GB_DEF_STP : BLOCKVECTOR
148      0281 1    [STEP_LEVELS, EVENTS$STEPPING_DESC_SIZE]
149      0282 1    FIELD-(EVENT$STEPPING_DESC_FIE[DS],
150      0283 1    dbg$gb_def_search: VECTOR [search_levels * search_lvl_size, BYTE].
151      0284 1    dbg$gb_def_define: VECTOR [define_levels * define_lvl_size, BYTE],
152      0285 1    dbg$gb_def_out: VECTOR [output_size, BYTE], ! DEFAULT OUTPUT CONFIG
153      0286 1
154      0287 1
155      0288 1    ++
156      0289 1    ****
157      0290 1    NOTE:
158      0291 1
159      0292 1    ALL initialization of addresses and filespec "strings" input to
160      0293 1    RMS user control blocks (FABs, RABs, etc.) MUST be done dynamically
161      0294 1    to maintain position independence. At compile time these addresses
162      0295 1    are relative to 0, but are relocated at run time since the debugger
163      0296 1    is brought in "behind" the user program.
164      0297 1    ****
165      0298 1    --
166      0299 1    ++
167      0300 1    declare the FAB and RAB blocks for terminal I/O.
168      0301 1    !--
169      0302 1
170      P 0303 1    dbg$gl_inpfab: SFAB (FAC=GET
171      P 0304 1    , MRS=no_of_inp_chars
```

```
172      0305 1
173      P 0306 1
174      P 0307 1
175      P 0308 1
176      P 0309 1
177      0310 1
178      P 0311 1
179      P 0312 1
180      0313 1
181      0314 1
182      0315 1
183      0316 1
184      0317 1
185      0318 1
186      P 0319 1
187      P 0320 1
188      P 0321 1
189      P 0322 1
190      P 0323 1
191      P 0324 1
192      0325 1
193      P 0326 1
194      0327 1
195      0328 1
196      0329 1
197      0330 1
198      0331 1
199      0332 1
200      0333 1
201      0334 1
202      0335 1
203      0336 1
204      0337 1
205      0338 1
206      0339 1
207      0340 1
208      0341 1
209      0342 1
210      0343 1
211      0344 1
212      0345 1
213      0346 1
214      0347 1
215      0348 1
216      0349 1
217      0350 1
218      0351 1
219      0352 1
220      0353 1
221      0354 1
222      0355 1
223      0356 1
224      0357 1
225      0358 1
226      0359 1
227      0360 1
228      0361 1

    dbg$gl_outpfab: $FAB (FAC=PUT
                           , MRS=tty_out_width
                           , RAT=<CRS
                           , SHR=<NIL>
                           )
    dbg$gl_inprab: $RAB (USZ=no of inp_chars
                           , ROP=<PMT>
                           )
    dbg$gl_outprab: $RAB (),
    ! Declare FAB and RAB blocks for LOG file
    dbg$gl_logfab: $FAB (RFM=VAR
                           , FAC=PUT
                           , FOP=<MXV>
                           , MRS=tty_out_width
                           , RAT=CR
                           , SHR=NIL
                           )
    dbg$gl_lograb: $RAB (ROP=<EOF>
                           ),
    ++
    | We'll give 20 trys for read error before take the exit.
    !--
    dbg$gl_readerr_cnt: INITIAL (0),
    ++
    | This the only bitvector.
    !!
    dbg$gl_context: BITVECTOR [context_bits],           ! context LONGWORD
    ! These are the global bytes.
    dbg$gb_set_module_flag: BYTE,                      ! TRUE during processing of SET MODULE.
    ! This changes the behavior of the
    ! lexers so that they allow module
    ! names that begin with numbers.
    dbg$gb_exc_bre_flag: BYTE,                         ! TRUE if we are in an EXCEPTION BREAK
    dbg$gb_go_arg_flag: BYTE,                          ! TRUE if there is an argument to GO
    dbg$gb_language: BYTE,                            ! HOLDS LANGUAGE INDEX
    dbg$gb_loc_type: BYTE,                            ! TELLS WHAT SORT OF END RANGE LOCATION
    dbg$gb_resignal: BYTE,                           ! BOOLEAN, TRUE IF RESIGNALING ALL EXCEPTIONS
    dbg$gb_take_cmd: BYTE,                           ! BOOLEAN, TRUE IF ANOTHER COMMAND WILL BE READ
    dbg$gb_tbit_ok: BYTE,                            ! TBITS ARE LEGAL
    dbg$gb_sym_status : BYTE,                         ! contains status of sym lookups.
    dbg$gb_no_globals : BYTE,                        ! replaces mc_global_locked flag.
    dbg$gb_keypad_input: BYTE,                        ! TRUE if we are doing keypad input
    dbg$gb_verb : BYTE,
    dbg$gb_radix:VECTOR[3,BYTE],                     ! Contains the radices specified
    ! in a SET RADIX[/OVERRR] command,
    ! or dbg$k_default if no radix
    ! override is in effect.
```

```
229      0362 1 | This can be indexed by three constants:  
230      0363 1 | dbg$b radix_input 0  
231      0364 1 | dbg$b radix_output 1  
232      0365 1 | dbg$b radix_output over 2  
233      0366 1 | Corresponding to the three kinds  
234      0367 1 | of radix settings (these are defined  
235      0368 1 | in DBGLIB). (This method of having  
236      0369 1 | a byte vector instead of three  
237      0370 1 | separate global bytes saves on  
238      0371 1 | link time, and should be used  
239      0372 1 | more extensively in this module.)  
240      0373 1 |  
241      0374 1 | dbg$gbUnhandledExc : VECTOR[10,BYTE] | dbg$gbUnhandledExc[0] is  
242      0375 1 | INITIAL(BYTE(REP 10 OF (0))) ! TRUE after an unhandled exception.  
243      0376 1 | ! This is a vector because we push  
244      0377 1 | ! the value on a call.  
245      0378 1 |  
246      0379 1 |  
247      0380 1 |  
248      0381 1 |++  
249      0382 1 | Global words.  
250      0383 1 |--  
251      0384 1 | dbg$gw_length, | a place for the parser to store a  
252      0385 1 | dbg$gw_locLength : INITIAL(0), | Length given in a verb modifier ty  
253      0386 1 | dbg$gw_gbLength : INITIAL(0), | Length given an override type spec  
254      0387 1 | dbg$gw_dflLength : INITIAL(0), | length given in a default type spe  
255      0388 1 |  
256      0389 1 |++  
257      0390 1 | Now refs to byte vectors. Don't confuse these with byte  
258      0391 1 | vectors.  
259      0392 1 |--  
260      0393 1 | dbg$gb_verptr : REF VECTOR[,BYTE], ! POINTER TO INPUT BUFFER FOR VERIFY  
261      0394 1 | dbg$gb_mod_ptr : REF VECTOR[,BYTE], ! POINTER TO CURRENT MODE LEVEL  
262      0395 1 | DBG$GB_STP_PTR : REF EVENTSSTEPPING_DESCRIPTOR, ! POINTER TO CURRENT STEP TYPE  
263      0396 1 | dbg$gb_search_ptr : REF VECTOR[,BYTE], ! Pointer to search settings  
264      0397 1 | dbg$gb_define_ptr : REF VECTOR[,BYTE], ! Pointer to define settings  
265      0398 1 |  
266      0399 1 |++  
267      0400 1 | REFS to more complicated (or more general)  
268      0401 1 | things than the above BYTE vectors.  
269      0402 1 | (The defn of the following 'types' is why  
270      0403 1 | DBGRST.BEG is included in this module.)  
271      0404 1 |--  
272      0405 1 |  
273      0406 1 | dbg$gb_logfsr : REF VECTOR[,BYTE], ! Resultant LOG filespec  
274      0407 1 | dbg$gb_logfse : REF VECTOR[,BYTE], ! Expanded LOG filespec  
275      0408 1 | dbg$gl_lognam : REF SNAME_DECL, ! LOG file NAM block  
276      0409 1 |  
277      0410 1 | ! Pointer to the current scope vector (CSP)  
278      0411 1 | DBG$GL_CSP_PTR : REF pth$pathname.  
279      0412 1 |  
280      0413 1 | DBG$GL_CISHEAD : REF CISLINK, ! Command input stream anchor  
281      0414 1 | DBG$GL_CIS_LEVELS : INITIAL(0), ! Number of levels of CIS  
282      0415 1 |  
283      0416 1 |++  
284      0417 1 | Normal longword vectors.  
285      0418 1 |--
```



```

: 343      0476 1
: 344      0477 1
: 345      0478 1
: 346      0479 1
: 347      0480 1
: 348      0481 1
: 349      0482 1
: 350      0483 1
: 351      0484 1
: 352      0485 1
: 353      0486 1
: 354      0487 1
: 355      0488 1
: 356      0489 1
: 357      0490 1
: 358      0491 1
: 359      0492 1
: 360      0493 1
: 361      0494 1
: 362      0495 1
: 363      0496 1
: 364      0497 1
: 365      0498 1 GLOBAL
: 366      0499 1
: 367      0500 1
: 368      0501 1
: 369      0502 1
: 370      0503 1
: 371      0504 1
: 372      0505 1 GLOBAL
: 373      0506 1
: 374      0507 1
: 375      0508 1 END
: 376      0509 0 ELUDOM

        ! as seen in TRANSFERSADDRESS
        ! DST record.

        ! Globals used for communication between the phases of the
        ! SET SOURCE command:
        dbgSgl_module : INITIAL(0),           ! Contains the module rst pointer
        dbgSgl_dirlst,                      ! Contains a pointer to the
                                            ! directory list.

        dbgSgl_log_buf,                    ! pointer to buffer containing LOG f

        ! Some globals used for communication between the phases of
        ! deposit_cmd:
        dbgSfloating_buffer : vector[30,byte],
        dbgSfloat_desc : BLOCK[8,BYTE],
        dbgSdbl_desc : BLOCK[8,BYTE],
        dbgSdyn_str_desc,                  ! pointer to descriptor for dynamic string
        dbgSdeposit_source : BLOCK[12, BYTE], ! A standard descriptor for the source
        dbgSdeposit_target : BLOCK[12, BYTE]; ! A standard descriptor for the target

        ! Globals used for dst and gst management.

        dbgSdst_begin_addr,                ! virtual address where DST begins.
        dbgSdst_end_addr,                 ! virtual address of last byte in DST.
        dbgSdst_next_addr,                ! virtual address where 'next' DST record begins.
        dbgSgsr_begin_addr,               ! virtual address where GST begins.
        dbgSGSR_next_addr : ref vector[,word]; ! virtual address of 'next' GST

        dbg$runframe : VECTOR[dbg$k_runfr_len/3 + 4, LONG] ! The current runframe
                                                INITIAL (REP (dbg$k_runfr_len/3 + 4) OF (0));

```

```

        .TITLE  DBGSTO
        .IDENT  \V04-000\

        .PSECT  DBG$PLIT,NOWRT, SHR, PIC,0

00 06 06 06 06 04 00 00 00 00 00 00 00 00 00 00000 P.**A: .BYTE
00 00 00 00 00 00 00 00 00 00 00 00 00 00 0000F
18 08 15 0A 09 10 00 1D 01 1C 10 05 04 00 00 0001E
0F 0E 02 02 02 02 02 02 02 02 0D 14 0C 0002D
01 01 01 01 03 03 03 03 03 13 00 17 19 16 0003C
01 01 01 01 01 01 01 01 01 01 01 01 01 01 0004B
07 07 08 08 08 08 08 08 00 01 11 18 12 1A 01 0005A
07 07 07 07 07 07 07 07 07 07 07 07 07 07 00069
00 00 00 00 00 00 00 07 07 07 07 07 07 07 00078

        .PSECT  DBG$GLOBAL,NOEXE, PIC,2
0000* 00000 DBG$GV_CONTROL:: .WORD <DBG$GL_SUP_OR_TEST:256>
00002          .BLKB 2

```

00000000# 00004 DBGSACCESS\_LIST::  
0001C DBGSGL\_MODRSTPTR2:: .LONG 0[6]  
00020 DBGSGB\_DEF MOD:: .BLKB 4  
00048 DBGSGB\_DEF STP:: .B[KB] 40  
00060 DBGSGB\_DEF SEARCH:: .B[KB] 24  
00066 DBGSGB\_DEF DEFINE:: .BLKB 6  
00068 DBGSGB\_DEF DEFINE:: .B[KB] 3  
0006B DBGSGB\_DEF OUT:: .BLKB 1  
0006C DBGSGL\_INPFB:: .B[KB] 3  
0006F DBGSGL\_INPFB:: .BLKB 1  
03 00070 DBGSGL\_INPFB:: .BYTE 3  
50 00071 .BYTE 80  
0000 00072 .WORD 0  
00000000 00074 .LONG 0  
00000000 00078 .LONG 0  
00000000 0007C .LONG 0  
00000000 00080 .LONG 0  
0000 00084 .WORD 0  
02 00086 .BYTE 2  
00 00087 .BYTE 0  
00000000 00088 .LONG 0  
00 0008C .BYTE 0  
00 0008D .BYTE 0  
00 0008E .BYTE 0  
02 0008F .BYTE 2  
00000000 00090 .LONG 0  
00000000 00094 .LONG 0  
00000000 00098 .LONG 0  
00000000 0009C .LONG 0  
00000000 000A0 .LONG 0  
00 000A4 .BYTE 0  
00 000A5 .BYTE 0  
0084 000A6 .WORD 132  
00000000 000A8 .LONG 0  
0000 000AC .WORD 0  
00 000AE .BYTE 0  
00 000AF .BYTE 0  
00000000 000B0 .LONG 0  
00000000 000B4 .LONG 0  
0000 000B8 .WORD 0  
00 000BA .BYTE 0  
00 000BB .BYTE 0  
00000000 000BC .LONG 0  
03 000C0 DBGSGL\_OUTPFAB:: .BYTE 3  
50 000C1 .BYTE 80  
0000 000C2 .WORD 0  
00000000 000C4 .LONG 0  
00000000 000C8 .LONG 0

C 2  
16-Sep-1984 02:39:14  
14-Sep-1984 12:17:50VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[DEBUG.SRC]DBGSTO.B32;1Page 9  
DE  
VC

00000000	000CC	.LONG	0
00000000	000D0	.LONG	0
0000	000D4	.WORD	0
01	000D6	.BYTE	1
20	000D7	.BYTE	32
00000000	000DB	.LONG	0
00	000DC	.BYTE	0
00	000DD	.BYTE	0
02	000DE	.BYTE	2
02	000DF	.BYTE	200
00000000	000E0	.LONG	0
00000000	000E4	.LONG	0
00000000	000E8	.LONG	0
00000000	000EC	.LONG	0
00000000	000F0	.LONG	0
00	000F4	.BYTE	0
00	000F5	.BYTE	0
0084	000F6	.WORD	132
00000000	000F8	.LONG	0
0000	000FC	.WORD	0
00	000FE	.BYTE	0
00000000	00100	.LONG	0
00000000	00104	.LONG	0
0000	00108	.WORD	0
00	0010A	.BYTE	0
00	0010B	.BYTE	0
00000000	0010C	.LONG	0
01	00110	DBG\$GL_INPRAB::	
		.BYTE	1
44	00111	.BYTE	68
0000	00112	.WORD	0
40000000	00114	.LONG	1073741824
00000000	00118	.LONG	0
00000000	0011C	.LONG	0
0000#	00120	.WORD	0[3]
0000	00126	.WORD	0
00000000	00128	.LONG	0
0000	0012C	.WORD	0
00	0012E	.BYTE	0
00	0012F	.BYTE	0
0084	00130	.WORD	132
0000	00132	.WORD	0
00000000	00134	.LONG	0
00000000	00138	.LONG	0
00000000	0013C	.LONG	0
00000000	00140	.LONG	0
00	00144	.BYTE	0
00	00145	.BYTE	0
00	00146	.BYTE	0
00	00147	.BYTE	0
00000000	00148	.LONG	0
00000000	0014C	.LONG	0
00000000	00150	.LONG	0
01	00154	DBG\$GL_OUTPRAB::	
44	00155	.BYTE	1
		.BYTE	68

D 2  
16-Sep-1984 02:39:14  
14-Sep-1984 12:17:50VAX-11 Bliss-32 V4.0-742  
DISK\$VMSMASTER:[DEBUG.SRC]DBGSTO.B32;1Page 10  
(1)

0000	00156	.WORD	0	
00000000	00158	.LONG	0	
00000000	0015C	.LONG	0	
00000000	00160	.LONG	0	
00000000	00164	.WORD	0[3]	
00000000	0016A	.WORD	0	
00000000	0016C	.LONG	0	
00000000	00170	.WORD	0	
00000000	00172	.BYTE	0	
00000000	00173	.BYTE	0	
00000000	00174	.WORD	0	
00000000	00176	.WORD	0	
00000000	00178	.LONG	0	
00000000	0017C	.LONG	0	
00000000	00180	.LONG	0	
00000000	00184	.LONG	0	
00000000	00188	.BYTE	0	
00000000	00189	.BYTE	0	
00000000	0018A	.BYTE	0	
00000000	0018B	.BYTE	0	
00000000	0018C	.LONG	0	
00000000	00190	.LONG	0	
00000000	00194	.LONG	0	
03	00198	DBG\$GL_LOGTAB::		
50	00199	.BYTE	3	
0000	0019A	.WORD	80	
00000002	0019C	.LONG	0	
00000000	001A0	.LONG	0	
00000000	001A4	.LONG	0	
00000000	001A8	.LONG	0	
0000	001AC	.WORD	0	
01	001AE	.BYTE	1	
20	001AF	.BYTE	32	
00000000	001B0	.LONG	0	
00	001B4	.BYTE	0	
00	001B5	.BYTE	0	
02	001B6	.BYTE	2	
02	001B7	.BYTE	0	
00000000	001B8	.LONG	0	
00000000	001BC	.LONG	0	
00000000	001C0	.LONG	0	
00000000	001C4	.LONG	0	
00000000	001C8	.LONG	0	
00	001CC	.BYTE	0	
00	001CD	.BYTE	0	
0084	001CE	.WORD	132	
00000000	001D0	.LONG	0	
0000	001D4	.WORD	0	
00	001D6	.BYTE	0	
00	001D7	.BYTE	0	
00000000	001D8	.LONG	0	
00000000	001DC	.LONG	0	
0000	001E0	.WORD	0	
00	001E2	.BYTE	0	
00	001E3	.BYTE	0	
00000000	001E4	.LONG	0	

01 001E8 DBG\$GL\_LOGRAB::  
44 001E9 .BYTE 1  
0000 001EA .WORD 0  
00000100 001EC .LONG 256  
00000000 001F0 .LONG 0  
00000000 001F4 .LONG 0  
0000# 001FB .WORD 0[3]  
0000 001FE .WORD 0  
00000000 00200 .LONG 0  
0000 00204 .WORD 0  
00 00206 .BYTE 0  
00 00207 .BYTE 0  
0000 00208 .WORD 0  
0000 0020A .WORD 0  
00000000 0020C .LONG 0  
00000000 00210 .LONG 0  
00000000 00214 .LONG 0  
00000000 00218 .LONG 0  
00 0021C .BYTE 0  
00 0021D .BYTE 0  
00 0021E .BYTE 0  
00 0021F .BYTE 0  
00000000 00220 .LONG 0  
00000000 00224 .LONG 0  
00000000 00228 .LONG 0  
00000000 0022C DBG\$GL\_READERR\_CNT::  
00230 .LONG 0  
00230 DBG\$GL\_CONTEXT::  
00234 .BLKB 4  
00234 DBG\$GB\_SET\_MODULE\_FLAG::  
.B[KB 1  
00235 DBG\$GB\_EXC\_BRE\_FLAG::  
.B[KB 1  
00236 DBG\$GB\_G0\_ARG\_FLAG::  
.BLKB 1  
00237 DBG\$GB\_LANGUAGE::  
.BLKB 1  
00238 DBG\$GB\_LOC\_TYPE::  
.B[KB 1  
00239 DBG\$GB\_RESIGNAL::  
.BLKB 1  
0023A DBG\$GB\_TAKE\_CMD::  
.BLKB 1  
0023B DBG\$GB\_TBIT\_OK::  
.BLKB 1  
0023C DBG\$GB\_SYM\_STATUS::  
.B[KB 1  
0023D DBG\$GB\_NO\_GLOBALS::  
.B[KB 1  
0023E DBG\$GB\_KEY\_AND\_INPUT::  
.BLKB 1  
0023F DBG\$GB\_Verb::  
.BLKB 1  
00240 DBG\$GB\_RADIX::  
.BLKB 5  
00243 .BLKB 1

00# 00244 DBG\$GB\_UNHANDLED\_EXC::  
.BYTE 0[10]  
0024E .BLKB 2  
00250 DBG\$GW\_LENGTH::  
.BLKB 4  
00000000 00254 DBG\$GW\_LCLNGTH::  
.LONG 0  
00000000 00258 DBG\$GW\_GBLNGTH::  
.LONG 0  
00000000 0025C DBG\$GW\_DFLTLENG::  
.LONG 0  
00260 DBG\$GB\_VRPTR::  
.BLKB 4  
00264 DBG\$GB\_MOD\_PTR::  
.B[KB] 4  
00268 DBG\$GB\_STP\_PTR::  
.B[KB] 4  
0026C DBG\$GB\_SEARCH\_PTR::  
.BLKB 4  
00270 DBG\$GB\_DEFINE\_PTR::  
.BLKB 4  
00274 DBG\$GB\_LOGFSR::  
.BLKB 4  
00278 DBG\$GB\_LOGFSE::  
.BLKB 4  
0027C DBG\$GL\_LOGNAM::  
.BLKB 4  
00280 DBG\$GL\_CSP\_PTR::  
.B[KB] 4  
00284 DBG\$GL\_CISHEAD::  
.BLKB 4  
00000000 00288 DBG\$GL\_CIS\_LEVELS::  
.LONG 0  
00000000# 0028C DBG\$REG\_VALUES::  
.LONG 0[17]  
00000000# 002D0 DBG\$REG\_VECTOR::  
.LONG 0[17]  
00314 DBG\$GL\_DIMENLST::  
.BLKB 40  
0033C DBG\$GL\_NEST\_STACK::  
.BLRB 100  
003A0 DBG\$GL\_NEST\_LEVEL::  
.BLRB 4  
003A4 DBG\$GL\_LIST::  
.BLKB 12  
003B0 DBG\$GL\_PARTBPTR::  
.BLKB 20  
003C4 DBG\$GL\_STK::  
.BLKB 480  
00000004 005A4 DBG\$GL\_ASCII\_LEN::  
.LONG 4  
005AB DBG\$GL\_BPTHEAD::  
.BLKB 4  
005AC DBG\$GL\_CMND\_RADIX::  
.BLKB 4  
005B0 DBG\$GL\_CURRENT\_PRIMARY::  
.BLKB 4

00000000 005B4 DBG\$GL\_MOVED\_DST\_LIST\_HEAD::  
005B8 DBG\$GL\_TYPE:: .LONG 0  
00000008 005BC DBG\$GL\_DFLTTYP:: .BLKB 4  
FFFFFFFFFF 005C0 DBG\$GL\_LÖCTYP:: .LONG 8  
00000000 005C4 DBG\$GL\_EDIT\_ENABLED:: .LONG -1  
FFFFFFFFFF 005C8 DBG\$GL\_GBLTYP:: .LONG 0  
005CC DBG\$GL\_GET\_LEX:: .LONG -1  
005D0 DBG\$GL\_HELP\_INPUT:: .BLKB 4  
005D4 DBG\$GL\_IND\_COM\_FILE:: .BLKB 4  
005D8 DBG\$GL\_LIS\_PTR:: .BLKB 4  
005DC DBG\$GL\_KEY\_TABLE\_ID:: .BLKB 2  
005E0 DBG\$GL\_KEYBOARD\_ID:: .BLKB 4  
005E4 DBG\$GL\_KEYW\_TBL:: .BLKB 4  
005E8 DBG\$GL\_LAST\_LOC:: .BLKB 4  
005EC DBG\$GL\_LAST\_VAL:: .BLKB 4  
00000000 005F0 DBG\$GL\_LIB\_SIGNAL\_ADDR:: .LONG 0  
00000000 005F4 DBG\$GL\_LIB\_STOP\_ADDR:: .LONG 0  
005F8 DBG\$GL\_NEXT\_LOC:: .BLKB 4  
00000000 005FC DBG\$GL\_OVRIDTYP:: .LONG 0  
00000000 00600 DBG\$GL\_SET\_SOURCE:: .LONG 0  
00000000 00604 DBG\$GL\_SET\_SOURCE2:: .LONG 0  
00608 DBG\$GL\_REDUC\_RT:: .BLKB 4  
0060C DBG\$GL\_STEP\_NUM:: .BLKB 4  
00610 DBG\$GL\_SYMHEAD:: .BLKB 4  
00000000 00614 DBG\$GL\_DLISLAST:: .LONG 0  
00618 DBG\$GL\_SEARCH\_VERB:: .BLKB 4  
00000000 0061C DBG\$GL\_TRANSFER\_ADDRESS:: .LONG 0  
00000000 00620 DBG\$GL\_MODULE:: .LONG 0  
00624 DBG\$GL\_DIRLIST:: .LONG 0

00628 DBGSGL\_LOG\_BUF:: .BLKB 4  
 .BLKB 4  
 0062C DBGSFLOATING\_BUFFER:: .BLKB 30  
 0064A .BLKB 2  
 0064C DBG\$FLOAT\_DESC:: .BLKB 8  
 00654 DBG\$DBL\_DESC:: .BLKB 8  
 0065C DBG\$DYN\_STR\_DESC:: .BLKB 4  
 00660 DBG\$DEPOSIT\_SOURCE:: .BLRB 12  
 0066C DBG\$DEPOSIT\_TARGET:: .BLRB 12  
 00678 DBG\$DST\_BEGIN\_ADDR:: .BLKB 4  
 0067C DBG\$DST\_END\_ADDR:: .BLRB 4  
 00680 DBG\$DST\_NEXT\_ADDR:: .BLKB 4  
 00684 DBG\$GSR\_BEGIN\_ADDR:: .BLKB 4  
 00688 DBG\$GSR\_NEXT\_ADDR:: .BLKB 4  
 00000000# 0068C DBG\$RUNFRAME:: .LONG 0[37]

DBG\$CHAR\_TABLE== P.AAA  
 .WEAK DBG\$GL\_SUP\_OR\_TEST

#### PSECT SUMMARY

Name	Bytes	Attributes
DBG\$GLOBAL	1824 NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON, PIC,ALIGN(2)	
DBG\$PLIT	128 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(0)	

#### Library Statistics

File	Symbols			Pages Mapped	Processing Time
	Total	Loaded	Percent		
\$255\$DUA28:[SYSLIB]LIB.L32:1	18619	30	0	1000	00:01.9
\$255\$DUA28:[DEBUG.OBJ]STRUDEF.L32:1	32	0	0	7	00:00.1
\$255\$DUA28:[DEBUG.OBJ]DBGLIB.L32:1	1545	41	2	97	00:01.9
\$255\$DUA28:[DEBUG.OBJ]DSTRECRDS.L32:1	418	0	0	31	00:00.3
\$255\$DUA28:[DEBUG.OBJ]DBGMSG.L32:1	386	1	0	22	00:00.3
\$255\$DUA28:[DEBUG.OBJ]DBGGEN.L32:1	150	30	20	12	00:00.3

DBGSTD  
V04-000

I 2  
16-Sep-1984 02:39:14    VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 12:17:50    DISK\$VMSMASTER:[DEBUG.SRC]DBGSTD.B32;1 (1)  
Page 15

: COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS\$:DBGSTD/OBJ=OBJ\$:DBGSTD MSRC\$:DBGSTD/UPDATE=(ENH\$:DBGSTD)

: Size:        0 code + 1952 data bytes  
: Run Time:    00:15.0  
: Elapsed Time: 00:18.0  
: Lines/CPU Min: 2036  
: Lexemes/CPU-Min: 27492  
: Memory Used: 113 pages  
: Compilation Complete

0095 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

